Low Power Wireless Shock Detection System

Dr. Michael Obara (michael.obara@navy.mil)

Nathan T. Brown (nathan.t.brown@navy.mil)

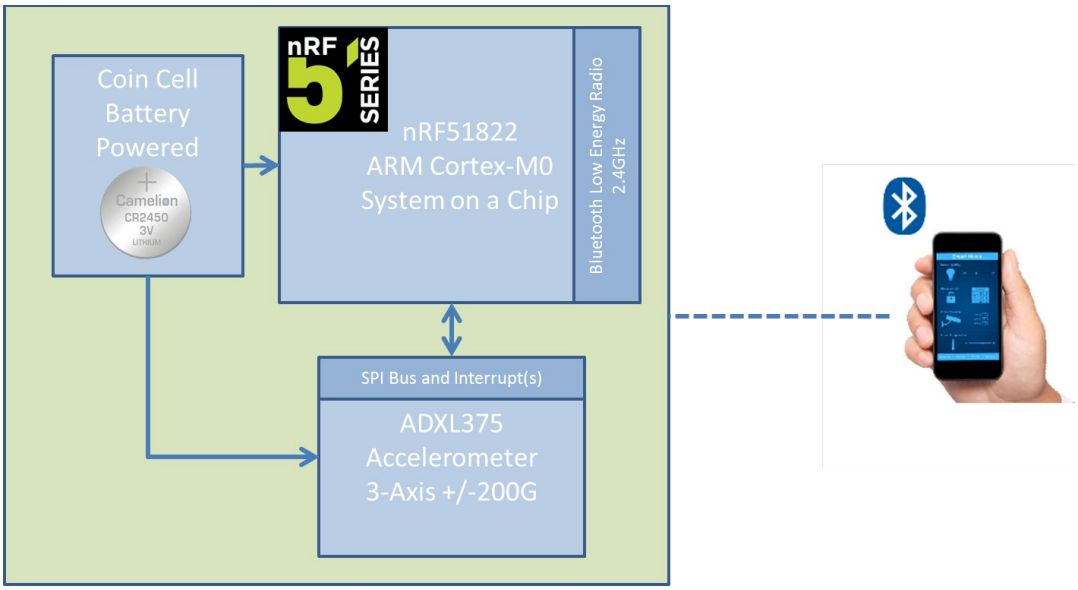
John Faella (john.faella@navy.mil)

Background:

The US Navy utilizes a wide variety of both acoustic and non-acoustic sensor systems. Many of these systems are deployed remotely and require significant operational lifetimes in order to meet mission objectives. To accomplish this, electronics designers must take into account not only requirements such as performance, cost and size but also power consumption. To reach these objectives, system level requirements must influence both the hardware and embedded software architecture. Critical decisions such as the type of battery, the microcontroller, and the sensor itself must be chosen so that when integrated, the cohesion enables a fluent and low power operation. In addition, the embedded firmware/software must have the ability to utilize different modes. Each mode specified a unique performance and power trade off. The mobility between these modes allows a dynamic operation which saves power without sacrificing effectiveness.

Project Details:

This project will employ a system which consists of two subsystems: a *remote* and a *host*. The *host* system will be an application running on a *Bluetooth Low Energy* (BLE) capable smartphone or tablet. It will communicate with the *remote* system allowing it to configure a shock threshold, duration before the threshold to capture and duration after the threshold to capture. The remote system once configured will go into an ultra-low power mode and await an event which meets its shock criteria. The *remote* system will consist of a *Nordic Semiconductor* Wireless System on a Chip (SoC) microcontroller, a coin cell battery for power and an *Analog Devices* three axis high G digital accelerometer. Initially the system will be prototyped with the use of provided evaluation/development kits. However, in the later stages a custom printed circuit board (PCB) will be designed and built.



Operational Flow:

The complete system will ultimately operate as follows:

1. The *host* application connects to the *remote* sensor device and configures it with the given parameters (i.e. 20G threshold with n milliseconds before and after the event). The *remote* device is then *deployed*.
2. The *remote* device configures its low power accelerometer to operate without microcontroller intervention (buffer samples in FIFO and look for transient shock events). In the event of a qualifying shock the accelerometer is programmed to assert one of its interrupt lines to wake the microcontroller out of its low power *sleep* mode. The microcontroller is then put to sleep.
3. If a transient is detected the micro is woken up and then it reads out the buffered data from the accelerometer. The *host* system is notified, and the data is then reliably transferred to the host where it can be reviewed and displayed. The *remote* system then goes back to sleep and continues to wait for shock events unless other instructions are given.

Requirements:

- Students:
  - Electrical Engineer: Build custom *remote* system circuit board after initial prototyping and support embedded software.
  - Computer Engineer: Write embedded software for microcontroller. MUST have strong understanding of C and computer architecture (Arduino does not count).
  - Computer Engineer: Write either iOS or Android application software for smartphone (MUST have access to smartphone or tablet which supports Bluetooth Low Energy/Smart. Android 4.4+ and iOS 5+).
- Development Environment:
  - nRF51822 SoC will be developed in Keil Microcontroller Development Kit (MDK) in the C language.

    - Smartphone/Tablet application will be developed in xCode (iOS) via the objective-C language or Eclipse (Android) via the Java language.
- *Remote* System:
    - Operate from coin cell battery for one month (assuming no shock events detected).
        - Derive power consumption from this requirement and battery capacity.
    - Interface with ADXL375 high G accelerometer.
        - Serial Peripheral Interface (SPI) communications.
        - Interrupt GPIO lines to wake up the microcontroller.
    - Implement custom BLE application layer structure to reliability communicate with *host* application over wireless link. Must thoroughly understand the BLE stack and implementation.
- *Host* System:
    - Reliably communicate with *remote* system (no data drops) over wireless BLE link.
    - Provide fluent and intuitive user interface (UI) in compliance with chosen architecture guidelines (iOS/Android).
    - Allow configuration of *remote* device and display of event data after a shock detection.